

AD-A245 948



NAVAL POSTGRADUATE SCHOOL
Monterey, California

2



THESIS

**NPSNET-MES:
SEMI-AUTOMATED FORCES
INTEGRATION**

by

Carl Patrick Cecil

September 1991

Thesis Co-Advisors:

Dr. Michael J. Zyda
David R. Pratt

Approved for public release; distribution is unlimited.

92 2 14 166

92-03977



REPORT DOCUMENTATION PAGE

1a. REPORT SECURITY CLASSIFICATION UNCLASSIFIED		1b. RESTRICTIVE MARKINGS	
2a. SECURITY CLASSIFICATION AUTHORITY		3. DISTRIBUTION/AVAILABILITY OF REPORT Approved for public release; distribution is unlimited	
2b. DECLASSIFICATION/DOWNGRADING SCHEDULE		5. MONITORING ORGANIZATION REPORT NUMBER(S)	
4. PERFORMING ORGANIZATION REPORT NUMBER(S)		7a. NAME OF MONITORING ORGANIZATION Naval Postgraduate School	
6a. NAME OF PERFORMING ORGANIZATION Computer Science Dept. Naval Postgraduate School	6b. OFFICE SYMBOL (if applicable) CS	7b. ADDRESS (City, State, and ZIP Code) Monterey, CA 93943-5000	
6c. ADDRESS (City, State, and ZIP Code) Monterey, CA 93943-5000		9. PROCUREMENT INSTRUMENT IDENTIFICATION NUMBER	
8a. NAME OF FUNDING/SPONSORING ORGANIZATION	8b. OFFICE SYMBOL (if applicable)	10. SOURCE OF FUNDING NUMBERS	
8c. ADDRESS (City, State, and ZIP Code)		PROGRAM ELEMENT NO.	PROJECT NO.
		TASK NO.	WORK UNIT ACCESSION NO.
11. TITLE (Include Security Classification) NPSNET-MES: SEMI-AUTOMATED FORCES INTEGRATION (U)			
12. PERSONAL AUTHOR(S) Cecil, Carl Patrick			
13a. TYPE OF REPORT Master's Thesis	13b. TIME COVERED FROM 08/89 TO 09/91	14. DATE OF REPORT (Year, Month, Day) 1991 September 26	15. PAGE COUNT 52
16. SUPPLEMENTARY NOTES The views expressed in this thesis are those of the author and do not reflect the official policy or position of the Department of Defense or the United States Government.			
17. COSATI CODES		18. SUBJECT TERMS (Continue on reverse if necessary and identify by block number)	
FIELD	GROUP	Path Planning, Networking, Autonomous Players	
19. ABSTRACT (Continue on reverse if necessary and identify by block number)			
<p>NPSNET-MES provides realistic semi-automated forces (SAF) for interactive play in the three-dimensional visual simulator, NPSNET. NPSNET-MES consists of two components. The first component is a path generation module that determines the SAF route and mission based upon the SAF controller input. This module generates multiple segment paths based upon obstacle avoidance. The second component is a vehicle controller module that uses the programmable network harness, NPSNET-NET, to multicast data packets via the Ethernet to control the SAF vehicles during the simulation. NPSNET-MES integrates SAF into an already existing network simulator such that no changes are necessary to the existing system, NPSNET. NPSNET-MES fulfills a real need in networking simulations to populate the battlefield simulation with semi-automated forces.</p>			
20. DISTRIBUTION/AVAILABILITY OF ABSTRACT <input checked="" type="checkbox"/> UNCLASSIFIED/UNLIMITED <input type="checkbox"/> SAME AS RPT. <input type="checkbox"/> DTIC USERS		21. ABSTRACT SECURITY CLASSIFICATION UNCLASSIFIED	
22a. NAME OF RESPONSIBLE INDIVIDUAL Dr. Micael J. Zyda and David R. Pratt		22b. TELEPHONE (Include Area Code) (408) 646-2305	22c. OFFICE SYMBOL CS/ZK

Approved for public release; distribution is unlimited.

**NPSNET-MES:
SEMI-AUTOMATED FORCES
INTEGRATION**

by

Carl P. Cecil
Major, United States Army
Bachelor of Science, United States Military Academy, 1979

Submitted in partial fulfillment
of the requirements for the degree of

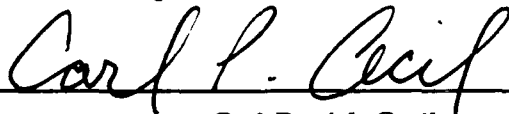
MASTER OF SCIENCE IN COMPUTER SCIENCE

from the

NAVAL POSTGRADUATE SCHOOL

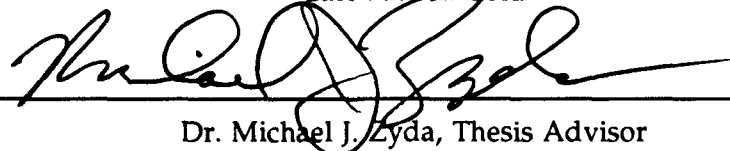
September 1991

Author:



Carl Patrick Cecil

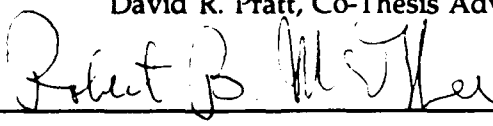
Approved by:



Dr. Michael J. Zyda, Thesis Advisor



David R. Pratt, Co-Thesis Advisor



Robert B. McGhee, Chairman
Department of Computer Science

ABSTRACT

NPSNET-MES provides realistic semi-automated forces (SAF) for interactive play in the three dimensional visual simulator, NPSNET. NPSNET-MES consists of two components. the first component is a path generation module that determines the SAF route and mission based upon the SAF controller input. This module generates multiple segment paths based on obstacle avoidance. The second component is a vehicle controller module that uses the programmable network harness, NPSNET-NET, to multicast data packets via the Ethernet to control the SAF vehicles during the simulation. NPSNET-MES integrates SAF into an already existing network simulator such that no changes are necessary to the existing system, NPSNET. NPSNET-MES fulfills a real need in networking simulation to populate the battlefield simulation with semi-automated forces.

Accession For	
NTIS CRA&I	<input checked="" type="checkbox"/>
DTIC TAB	<input type="checkbox"/>
Unannounced	<input type="checkbox"/>
Justification	
By	
Distribution/	
Availability Codes	
Dist	Avail and/or Special
A-1	



ACKNOWLEDGEMENTS

I take this opportunity to thank the people who provided invaluable assistance or inspiration for this project. I thank Professor Mike Zyda for his taking me on as a thesis student. My sincere thanks to Dave Pratt for his help and never ending patience with me. Thanks to the guys in the Graphics Lab who lent a helping hand to get me past some program bugs. Thanks to the Naval Postgraduate School technical staff for keeping the systems going when it really matters.

Finally, I thank my wife and children for their understanding when I was spending those late evenings in the Graphics Lab trying to get NPSNET-MES off the ground.

TABLE OF CONTENTS

I. INTRODUCTION	1
A. BACKGROUND	1
B. NPSNET	2
C. MOTIVATION FOR NPSNET SAF INTEGRATION	3
D. RESEARCH QUESTIONS	3
E. ORGANIZATION	4
II. SURVEY OF PREVIOUS WORK	5
A. INTRODUCTION	5
B. PREVIOUS VEHICLE SIMULATION AND AUTONOMY SYSTEMS	5
1. A Prototype Simulation System for Combat Vehicle Coordination and Motion Visualization.	5
2. A Computer Simulation Study of the Rule-Based Control of an Autonomous Vehicle.	6
3. An Autonomous Platform Simulator (APS).	7
C. PREVIOUS PATH-PLANNING ALGORITHMS	7
1. Obstacle Avoidance and Navigation in the Real World by a Seeing Robot Rover.	7
2. Real-Time Mission and Trajectory Planning.	8

D.	CONCLUSION	8
III.	PROBLEM DESCRIPTION AND METHODOLOGY	10
A.	INTRODUCTION	10
B.	PROBLEM DESCRIPTION	10
C.	BEST METHOD TO INTEGRATE AND REPRESENT THE SAF IN NPSNET	11
D.	PATH GENERATION MODULE	12
1.	Circle World	12
2.	Computational versus Apriori Path Planning	14
E.	VEHICLE CONTROLLER MODULE	16
F.	CONCLUSION	17
IV.	SYSTEM DESCRIPTION	19
A.	TERRAIN DATABASE	19
B.	MOBILITY EXPERT SYSTEM SIMULATOR	20
1.	NPSNET-MES Capabilities	20
2.	SAF Controller Input	20
3.	Network Communications	21
a.	NPSNET-NET	21
b.	NPSNET-MES Networking	21
4.	Module Descriptions	22

a.	Path Generation Module	22
b.	Vehicle Controller Module	23
C.	SUMMARY	23
V.	NPSNET-MES RESULTS	25
A.	PATH GENERATION MODULE	25
B.	VEHICLE CONTROLLER MODULE	27
C.	SUMMARY OF RESULTS	29
VI.	SUMMARY AND CONCLUSIONS	31
A.	LIMITATIONS	31
1.	Path Generation Module	31
2.	Vehicle Controller Module	32
B.	AREAS FOR FURTHER RESEARCH	33
C.	SUMMARY	34
D.	CONCLUSIONS	35
APPENDIX A.	USER'S GUIDE	36
A.	PATH GENERATION MODULE	36
B.	VEHICLE CONTROLLER MODULE	37
APPENDIX B.	DATA STRUCTURES AND FUNCTION DESCRIPTIONS	39

A.	DATA STRUCTURES	39
1.	Path Generation Module	39
2.	Vehicle Controller Module	40
B.	FUNCTION DESCRIPTIONS	40
1.	Path Generation Module	40
2.	Vehicle Controller Module	41
	LIST OF REFERENCES	42
	INITIAL DISTRIBUTION LIST	43

I. INTRODUCTION

A. BACKGROUND

The major objective of this work is to develop a prototype module for the semi-automated forces (SAF) in NPSNET. NPSNET is an interactive low-cost, three-dimensional visual simulator using a network of graphics and non-graphics workstations [Zyda, 1991, p.361]. NPSNET employs SAF to introduce sufficient numbers of unmanned players into the system to make the simulation more challenging and exciting. One component of SAF is a mobility expert system (MES). The mobility expert system integrates a modified breadth-first search technique of path planning into the NPSNET SAF.

The NPSNET mobility expert system (NPSNET-MES) consists of two components. The components are the path generation module and the vehicle controller module. The NPSNET-MES user specifies a path for the lead vehicle and the prototype system generates a path for the lead vehicle as well as the combat formation of vehicles. The NPSNET-MES vehicle controller module uses the programmable network harness, NPSNET-NET, to multicast packets via the Ethernet to control the vehicles during the simulation.

B. NPSNET

NPSNET is a real-time, 3D visual simulation system capable of displaying vehicle movement over the ground or in the air [Zyda, 1991, pp.361-363]. Displays show cultural features such as roads, buildings, soil types and elevations. The system is capable of environmental effects such as fog or haze. NPSNET supports a full complement of vehicles, houses, trees, signs, watertowers, and much more. The user selects a vehicle to drive via mouse selection on a command and control screen. Vehicle movement is controlled by a six degree of freedom SpaceBall or button/dial box. Vehicles interact with other vehicles. Up to 500 vehicles can be in motion at anytime. Other vehicles are controlled by a script file, or are driven interactively from other workstations, over the communications medium via packets multicast over Ethernet. The NPSNET system explores a virtual world using a readily available graphics workstation, the Silicon Graphics IRIS workstation.

There are a variety of applications for this system, including training, planning, gaming and other purposes where the introduction of the physical player can be too hazardous or too expensive. The system is an ideal platform for the visualization of complex 3D environments, such as battlefield training simulator. The high degree of interaction and low cost make NPSNET an effective method for small unit leadership and coordination training as well as mission and route planning. NPSNET validates and provides 3D playback for a combat model.

C. MOTIVATION FOR NPSNET SAF INTEGRATION

NPSNET uses randomly guided vehicles to populate the battlefield. These vehicles have very little intelligence and are only capable of firing back at an attacker or running away. The importance of populating the battlefield with combat formations that act semi-autonomously gave rise to this project. It is not enough to have random vehicles moving about the battlefield without a mission. The NPSNET mobility expert system allows one or more combat formations to intelligently maneuver across the battlefield.

The Department of Defense is looking for new and innovative means to train our forces at minimal expense. NPSNET provides a low-cost medium to train and test combat skills. SAF affords realistic friend or foe forces to populate the battlefield.

NPSNET is an ongoing research prototype [Zyda, 1991, pp.362-363]. The following are other topics currently being researched: SIMNET database initial display, ITD terrain database utilization, hierarchical data structures for real-time display generation, software structure for world modeling and interaction, SIMNET network integration, physically-based modelling for displaying the results of interaction, 3D icon production, representation and abstraction, and aural cues for 3D visual simulation.

D. RESEARCH QUESTIONS

This work addresses the following major research questions:

- * What is the best approach to incorporate SAF into NPSNET?
- * What is the best method to integrate and represent NPSNET-MES in the NPSNET system?

The remainder of this work examines these questions to show how to best implement NPSNET-MES into NPSNET.

E. ORGANIZATION

Chapter II reviews previous work relating to autonomous forces. Chapter III discusses the problem description and the approach used to integrate SAF into NPSNET. Chapter IV examines the best method used to integrate and represent MES in NPSNET. Chapter V addresses system results and provides an examination and analysis of performance characteristics. Chapter VI denotes the system limitations, proposes future thesis work on NPSNET-MES systems, and contains a summary of conclusions and recommendations. Appendix A is a user's guide for the prototype system. Appendix B is a list the program's data structures and function descriptions.

II. SURVEY OF PREVIOUS WORK

A. INTRODUCTION

The majority of autonomous vehicle simulators share several common characteristics. First, the simulators are computer based. Second, they rely heavily on computer visualization because of the high cost of building the platform [McConkle, 1988, p.3]. The use of autonomous vehicle simulators for semi-automated forces in computer simulations is not new, but this research adds a new dimension. This chapter provides the background information for the NPSNET-MES research.

B. PREVIOUS VEHICLE SIMULATION AND AUTONOMY SYSTEMS

1. *A Prototype Simulation System for Combat Vehicle Coordination and Motion Visualization.*

McConkle and Nelson developed a prototype rule-based command and control system for autonomous tactical vehicles in a graphics simulation [McConkle, 1988, pp.12-39 and Zyda, 1990, pp.321-333]. This system employs rule-based actions to model the expert system's autonomous behavior. The system allows multiple vehicle formations to maneuver in the graphics simulation. Each vehicle has simulated vision, a pilot, and steering and velocity control mechanisms. The vehicles can act independently or as a part of a tactical autonomous unit. The prototype uses a network to communicate the vehicle controller commands from the TI/Explorer LISP Machine to the visual display on the

Silicon Graphics IRIS 2400-Turbo workstation. A major weaknesses with the prototype is the requirement for a LISP machine to control each autonomous vehicle. McConkle and Nelson's system demonstrates autonomous vehicle actions, but is extremely computer hardware intensive and is very difficult to operate.

2. A Computer Simulation Study of the Rule-Based Control of an Autonomous Vehicle.

MacPherson's Autonomous Underwater Vehicle (AUV) simulation study tests new ideas and algorithms for the control of the AUV [MacPherson, 1988, pp.1-17]. A major contribution of his work is the introduction of the mission template that allows for the selection of an assortment of mission packages. MacPherson's work emphasizes the development of a family of programs that define AUV missions that are executable in a computer graphics simulation. MacPherson configures the simulator hierarchical system in three levels. At the highest level is the mission level that interfaces with the user, translating mission command into general guidance. The next level of control is the guidance level, that converts the general guidance into rule-based path-planning/obstacle avoidance control. The lowest level is the execution level that conducts sensor management and actual control of the vehicle. The simulator uses an Ethernet network to communicate the vehicle controller commands from the TI/Explorer LISP Machine to the visual display on the Silicon Graphics IRIS 2400-Turbo workstation.

3. An Autonomous Platform Simulator (APS).

Shannon and Teter's APS is a low cost testbed for the study of real-time path planning algorithms [Shannon, 1989, pp.1-17]. APS is a bridge between theoretical AI path-planning problems and applied research because it provides a realistic environment to achieve concrete results using AI path-planning algorithms. The APS solution is adequate for general purpose path-planning, but is too slow for the desired purpose of real-time path-planning. The APS path planning algorithm takes many factors into account such as the impact of terrain slope data. APS uses the wavefront algorithm to select the best route. The system uses a Symbolics 3600 LISP Machine to control the vehicle while displaying the image on the Silicon Graphics IRIS/4D-70GT graphics workstation. APS uses an Ethernet network to communicate between the path-planner and the image display computer.

C. PREVIOUS PATH-PLANNING ALGORITHMS

1. Obstacle Avoidance and Navigation in the Real World by a Seeing Robot Rover.

Moravec uses a path-planning algorithm that depicts the world in terms of circular objects [Moravec, 1980, pp.53-63]. The path-planner is a high-level component of the guidance system of a mobile robot that is programmed to navigate through the halls of a lab. The Seeing Robot Rover represents obstacles and the moving objects along the path as circles in 2D world. The robot finds a shortest path from start to finish using an apriori breadth-first graph search. This method is expensive in terms of time and space.

Moravec adapts this optimal solution to return an approximated path by adding a heuristic of reducing the search space. The modified search algorithm performs better than the full search space algorithm with the Seeing Robot Rover because it requires real-time path data.

2. Real-Time Mission and Trajectory Planning.

Beaton, Adams and Harrison designed a hierarchical planner that develops missions and provides navigational control for autonomous vehicles [Beaton, 1987, pp.1954-1959]. The trajectory planner creates multiple, near-optimal mission paths and estimates the cost in terms of energy, time, and lethality. The mission planner uses this information to conduct its mission planning and determine the optimal path and subgoals along the way. The system uses a modified A* search algorithm to improve efficiency and speed of the graph search.

D. CONCLUSION

The previous vehicle simulation and display systems require path-planning guidance that returns real-time data. The importance of real-time information for an autonomous vehicle is crucial. Real-time data keeps the autonomous vehicles from self-destructing. Therefore, good results are necessary, but not at the expense of getting perfect path data late. Some previous path-planning algorithms reduce the search space to achieve near optimal paths and improve real-time efficiency. Moravec's path planning algorithm uses a circle world much like NPSNET-MES. Moravec also relaxes the path-planning algorithm to achieve real-time performance using an approximated optimal path rather

than the actual optimal path. In NPSNET-MES like some of these systems, the trade-off for a good fast solution is also made over a perfect path solution.

III. PROBLEM DESCRIPTION AND METHODOLOGY

A. INTRODUCTION

The mobility expert system is an excellent choice for semi-automated forces to enhance the interactive simulation. NPSNET-MES in NPSNET plays a vital role in making the simulation a more viable training instrument. Without the introduction of semi-automated forces, the user can only have a limited number of intelligent players capable of networking into the simulation.

This chapter reviews the algorithms to implement the NPSNET-MES prototype system that make it compatible with the NPSNET system. NPSNET-MES has two major components. The first component is the path generation module. The path generation module allows the SAF controller to specify the number of combat formations and the number of vehicles in each formation as well as a path for each formation. The other component, the vehicle controller module, uses the network harness to multicast data packets of the paths generated by the path generation module onto the network for use by NPSNET.

B. PROBLEM DESCRIPTION

One of the major objectives of this work is to determine the best approach to integrate semi-automated forces into the already existing NPSNET system. The following are the minimum capabilities semi-automated forces:

- The SAF controller specifies a path that includes start and goal points with possible way points along the route.
- The SAF must negotiate all known obstacles without hitting them in a relatively optimal path.
- The SAF vehicles within a SAF formation must follow the lead SAF vehicle such that they maintain relative positions and do not collide with each other.
- The SAF controller specifies the number of combat formations as well as the number of vehicles, speed and type of each combat formation.
- When a SAF vehicle is killed, it no longer moves.

NPSNET-MES integrates the SAF into the existing NPSNET without major overhaul of the system. Once the SAF controller determines the SAF prerequisite information, NPSNET-MES makes that information available to the NPSNET for use during the simulation. These basic considerations drive the requirements for the NPSNET-MES prototype system. The remainder of this chapter describes the conceptual ideas that NPSNET-MES uses to resolve these questions.

C. BEST METHOD TO INTEGRATE AND REPRESENT THE SAF IN NPSNET

The integration of SAF into NPSNET is one of the major tasks for this project. To get the desired results, NPSNET-MES is designed to act in a stand alone mode. This means that NPSNET-MES integrates the SAF into NPSNET by using the existing NPSNET-NET, a set of programmable network harness routines. The problem separates into two distinct subsets. First part is to design semi-automated forces that can navigate and travel a specified path. The second part is to transmit the information generated by the first part.

NPSNET-MES is a 2D map/interface for SAF vehicle placement and route selection. The SAF controller is able to control the SAF parameters, such as speed, number of SAF formations, number of vehicles in each SAF formation, type of SAF vehicles, and general route selection. NPSNET-MES stores this information in a file ready for use by the second program at any time the SAF controller wishes to generate SAF during the simulation.

The second part of NPSNET-MES is a vehicle controller that uses NPSNET-NET. All communications between NPSNET-MES and NPSNET are through the network using the network harness, NPSNET-NET, that facilitates network communications between NPSNET and other modules [Zyda, 1991, pp.362-363]. The vehicle controller program ensures that SAF vehicles stay on track with the paths generated by the first program.

D. PATH GENERATION MODULE

The path generation module is a 2D map/interface that the SAF controller performs SAF vehicle placement and route selection. The path selection criteria for this module is not an optimal path, rather it is a relatively simple path that is found quickly. This module generates a path based on apriori obstacle information using a circle world.

1. Circle World

The path generation module uses a circle world to describe the known obstacles on the simulation terrain map. All obstacles such as houses, buildings, trees, and others are abstracted as circles using the radius from the center of the obstacle to the most distant vertex (Figure 1). Using circles in lieu of the actual obstacle boundaries

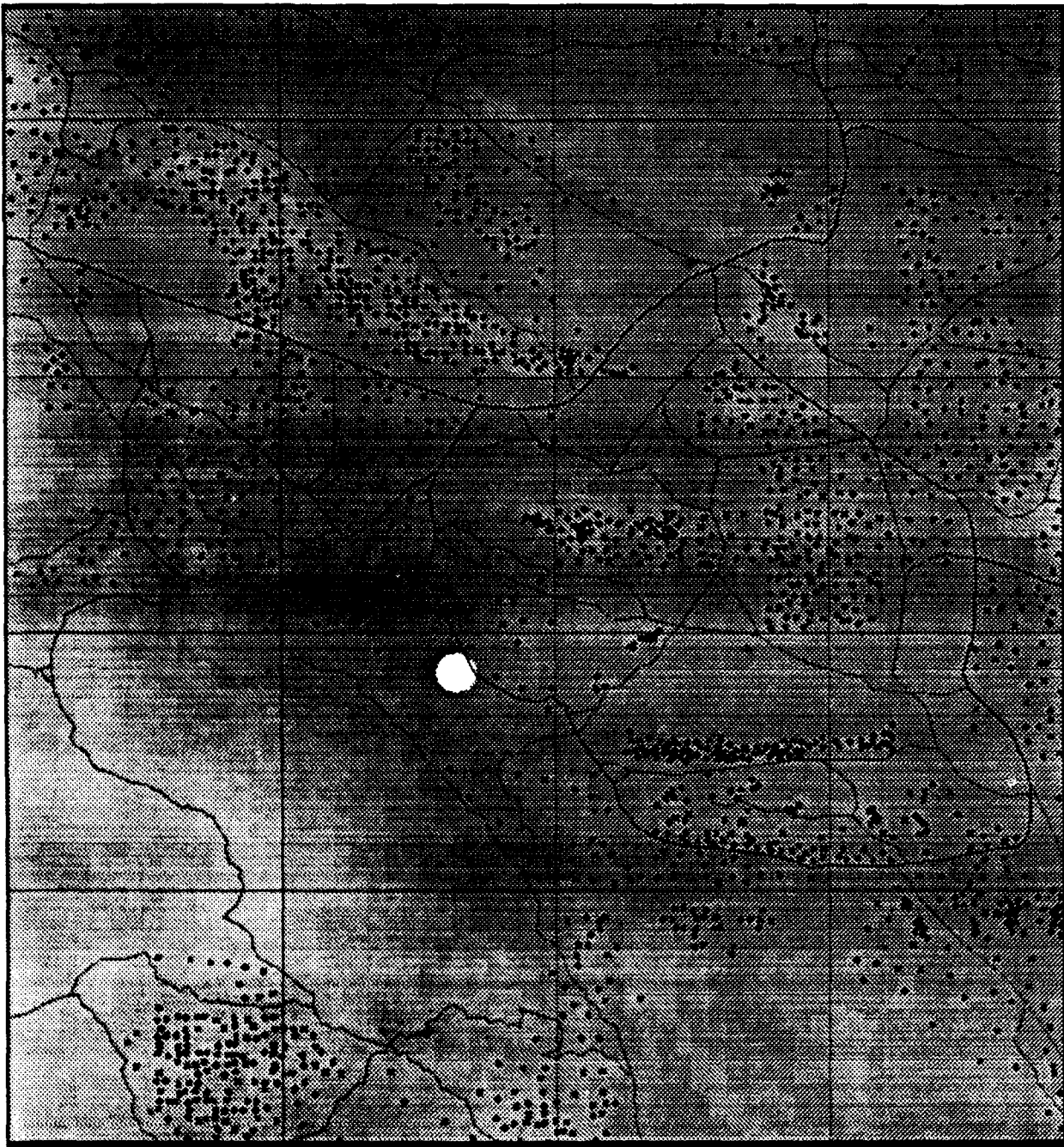


Figure 1. Circle World

saves calculation time since only two tangents are checked instead of each obstacle vertex.

2. Computational versus Apriori Path Planning

The path generation module's path generation algorithm uses a modified breadth-first search of a bounding box rather than the more traditional artificial intelligence approach of apriori generated paths because it more efficient and less complex. The computational approach searches the bounding box shooting a line between the start and goal to determine if the goal is visible from the start. If the path has an obstacle then the path finder is called recursively until a path is found around obstacles enroute to the goal (Figure 2). There are thousands of obstacles in the obstacle data file. The NPSNET-MES path generation module algorithm bounds the search area using the start and goal points to limit the search within that box (Figure 3). An apriori path generation produces paths for the entire database requiring a longer amount of time and more memory to store those paths for quick access. The recursive path planner grows in a linear fashion versus a non-linear growth for the more traditional apriori method. For example, by adding one more obstacle the computational search grows by four, whereas the apriori search grows by 12 (Figure 4).

Path Generation Module

user inputs path start and goal points
determine the path direction
establish bounding box using start & goal points to limit search

```
procedure find path( start, goal )  
  
  loop x and z in mapgrid matrix  
    if intersection and obstacle not in list  
      find closest obstacle tangent  
    endif  
  
    set boolean obstacle in path True  
    procedure find path( start, obstacle tangent)  
    procedure find path( obstacle tangent, goal )  
  endloop  
  
  if obstacle not in path  
    add path segment  
  endif  
  
endprocedure find path
```

Figure 2. Path Planning Algorithm

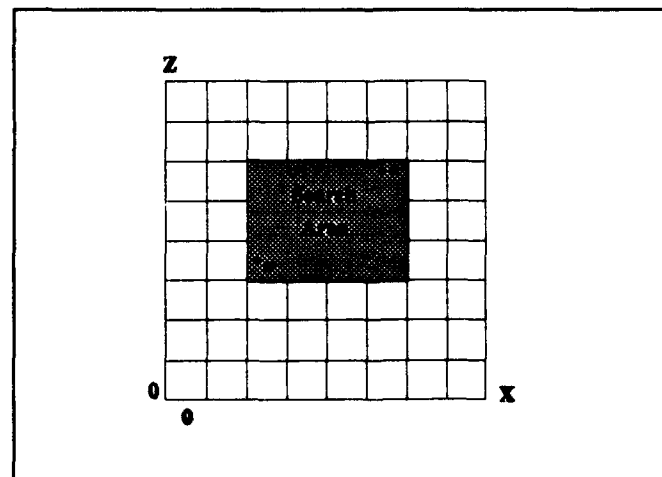


Figure 3. Bounding Box Search

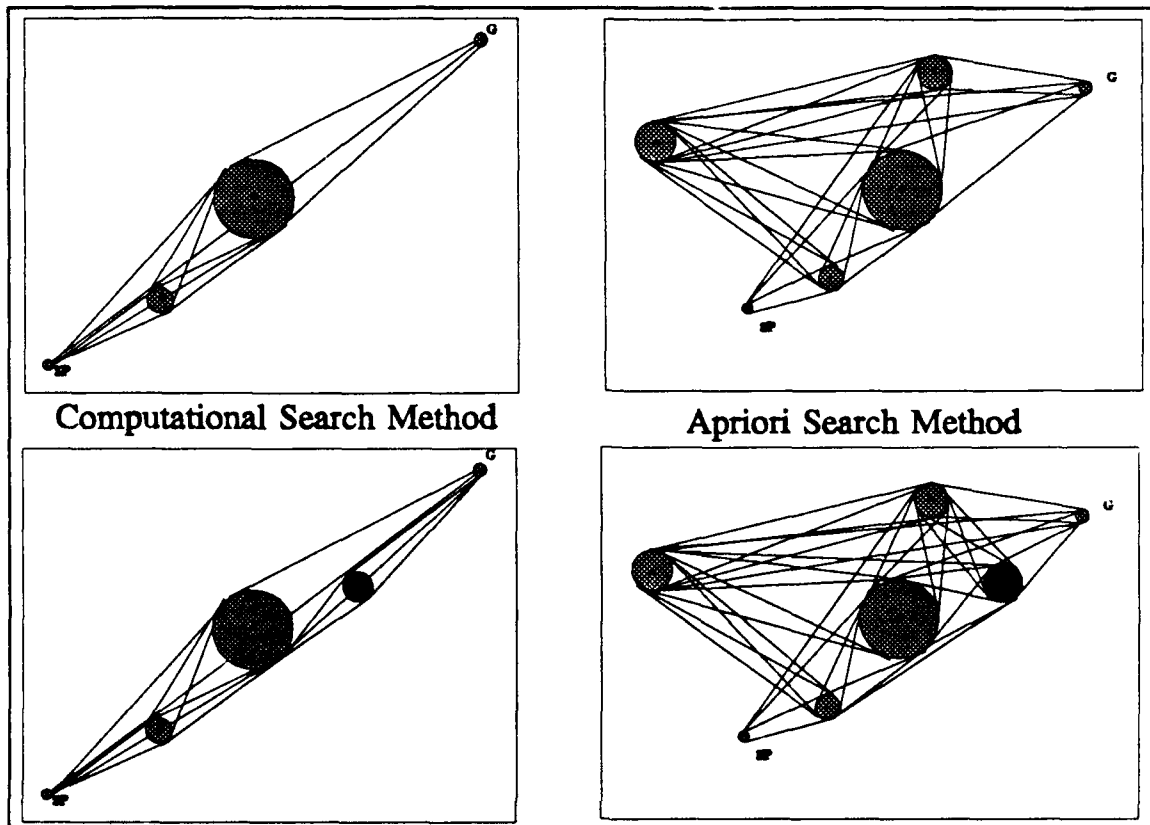


Figure 4. Computational vs Apriori Search

E. VEHICLE CONTROLLER MODULE

The vehicle controller module is the interface for NPSNET-MES with NPSNET. This program places the paths generated in the path generation module in a sorted linked list by ascending order of time. A path point time is a running total time for the vehicle from the start up to that point. Using the system clock to maintain relative time, the paths are taken off of the priority list when the delta-time exceeds the path point time. The NPSNET-NET sends updated messages reflecting the new vehicle position, direction,

and speed to NPSNET. NPSNET receives the path data and the SAF vehicles respond to the vehicle controller commands (Figure 5).

Vehicle Controller Module

read the path generation file

place vehicle/convoy path points into linked list sorted by time

add process to the network

establish a relative start time from system clock

assign current pointer to head pointer of sorted linked list

while not end of list

 get new time from system clock

 delta time is new time - start time

 if current path time \leq delta time

 get packets from network

 if vehicle is alive

 send data point to network

 increment the pointer

 endif

 endif

endloop

detach process from the network

Figure 5. Vehicle Controller Algorithm

F. CONCLUSION

NPSNET-MES provides a relatively efficient solution to finding a good path for the SAF vehicles. The path found by the path generation module does not attempt to find the best solution only a good solution, since the human that it emulates usually only finds a good solution when conducting path planning. The vehicle control module provides the

necessary interface between NPSNET-MES and NPSNET so that the SAF forces travel much as they do in real life.

IV. SYSTEM DESCRIPTION

This chapter details how the system and its algorithms are implemented. NPSNET-MES operates in a database of 2500 square kilometers located in Monterey County, California. The system allows the SAF controller to determine the make-up of various SAF formations for introduction into an NPSNET simulation. NPSNET-MES uses NPSNET-NET to multicast data packets over the Ethernet. Appendix B contains the details on the program's data structures and function descriptions.

A. TERRAIN DATABASE

NPSNET-MES uses the same terrain database as NPSNET, a 50 km by 50 km grid containing Ft Hunter-Liggett, California. The elevation data is stored in a mapgrid matrix by its X and Z coordinates. Post spacings in the mapgrid matrix are 125 meters by 125 meters. The database is displayed by elevation with the highest points shaded darkest and the lowest lightest.

Cultural features as well as terrain features are displayed on the 2D map in the NPSNET-MES path generation module. Cultural features include man-made objects like buildings, houses, and telephone poles. Terrain features include trees, shrubs, rocks, etc. Additionally, NPSNET-MES displays roads and railroads from a separate file.

B. MOBILITY EXPERT SYSTEM SIMULATOR

1. NPSNET-MES Capabilities

NPSNET-MES has the following capabilities:

- Find a path or paths for multiple vehicles with multiple path segments and controller entered speed for each path segment.
- Replot the path.
- Combat formation vehicles maintain relative position during movement.
- Display 2D map of the simulation area.
- Display projected and actual combat formation paths on 2D map.
- Multicast data packets to all network stations so that multiple IRIS workstations can monitor the SAF movement.
- Stop dead SAF vehicles from moving.

2. SAF Controller Input

The SAF controller determines the type, number, and configuration of the semi-automated forces that are introduced into the NPSNET simulation using NPSNET-MES. The SAF controller generates the SAF paths at anytime as long as there is access to an IRIS workstation. The controller executes the path file at anytime during the execution of the NPSNET simulation as long as there is access to a terminal on the network. Therefore, the SAF controller generates several SAF missions for execution at various stages of the simulation.

The SAF controller sets several variables to determine the SAF mission and actions:

- Designate the desired direction of movement to include start and goal points.
- Designate intermediate points along the path.
- Set MES vehicle types.
- Set MES combat formation size.
- Set MES combat formation lead vehicle speed for each path segment.
- Label a unique path file by name for each SAF mission from the command line.
- Call up a unique SAF path file by name at the unix command line prompt for the vehicle controller module.

3. Network Communications

a. NPSNET-NET

NPSNET-NET, the programmable network harness, is a separate program that runs in the background and executes the network commands. NPSNET-NET's **netdemon** program allows the user to enter a process onto the network. The **netdemon** program takes care of starting up the network, passing messages/packets over the network, handling interrupts, and managing the network. NPSNET-NET's **userproc.c** file provides functions to work hand-in-hand with the **netdemon** program.

b. NPSNET-MES Networking

NPSNET-MES uses several of the available functions from NPSNET-NET's **userproc.c** file to facilitate the networking of the SAF paths. NPSNET-MES uses

addproctonet, **getpackets**, **sendupdatemess**, and **detachfromnet** functions to get onto the network, pass message traffic and quit the network.

4. Module Descriptions

a. Path Generation Module

NPSNET-MES conducts the majority of its work in this module by creating the paths for execution by the SAF vehicles. The **main** function primarily sets up the initial 2D display map and initializes the input devices. The **main** calls **readfiles** and **readmodels** that return data structures used by the rest of the program. Once **find_and_write_path** is called by main, the program does not return to **main** until **pathfile.dat** file is written to memory. The function **find_and_write_path** calls the **display_path_info** function so that the user can see the variables that are input. The **draw_route** functions allows the user to input a desired path with intermediate rendezvous points as well as a speed for each path segment. The user can make modifications to the displayed path. Once the path is set then **convoy_disposition** determines the start, goal, and intermediate points for the vehicles within the combat formation. **Path_finder** generates path points and directions for the entire path. **Calculate_distance** and **calculate_speed_and_time** use the **full_path** array data created by **path_finder** to determine the remaining critical information for safe path planning. The most important aspects of this module are in Figure 6.

b. Vehicle Controller Module

Though the majority of the work takes place in the path generation module, the vehicle controller module has a critical job of taking the path data and successfully transmitting it to the SAF vehicles in the NPSNET simulation. The **main** function reads the path file and calls **add_to_sorted_list** to place the data in an ascending order linked list. **Addproctonet** is a call to NPSNET-NET that adds the process to the network. When delta-time grows to equal or exceed the path point time then **main** calls NPSNET-NET's **getpackets** function to determine if the vehicle is alive. If alive, then **main** calls NPSNET-NET's **sendupdatemess** function. **Sendupdatemess** sends the next vehicle path point data to NPSNET. Upon exhaustion of the priority queue of path points, NPSNET-NET's **detachfromnet** function deletes the process from the network. Figures 7 illustrates program flow control for the vehicle controller module.

C. SUMMARY

NPSNET-MES is a capable system for the integration of SAF into visual simulations. This system makes the networking simulation more challenging and exciting. The next chapter describes the results that NPSNET-MES achieves in various tests.

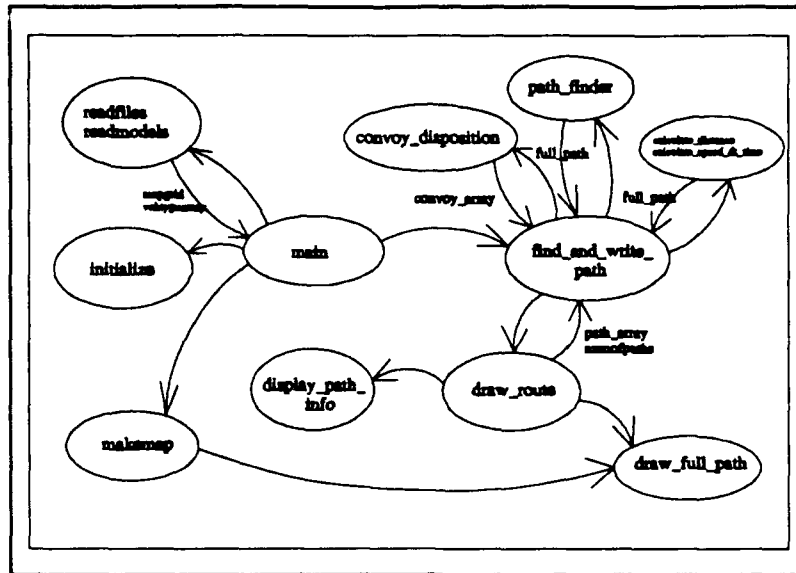


Figure 6. Path Generation Module Flow Control

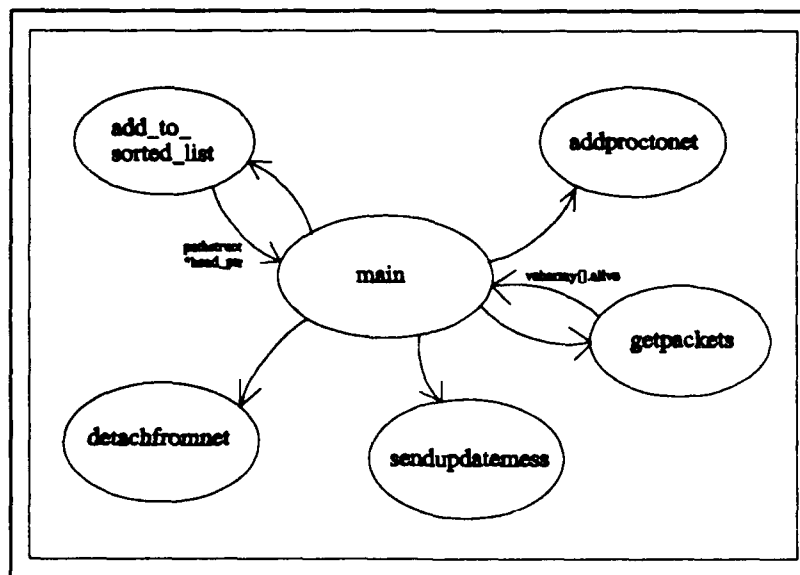


Figure 7. Vehicle Controller Module Flow Control

V. NPSNET-MES RESULTS

NPSNET-MES achieves most of the initial research goals. The system provides a realistic friend or foe force on the simulation battlefield. NPSNET-MES effectively integrates semi-automated forces into NPSNET. After many simulation tests, NPSNET-MES proves highly reliable in finding a realistic path and keeping the combat formations on that path. This system is a prototype for research, therefore it has many potential capabilities that can be added at a later time that.

A. PATH GENERATION MODULE

The path generation module achieves the goals stated in Chapter IV. This module produces a near optimal path that the SAF controller specifies the start, goal, and intermediate points along the route. The path planner returns paths for multiple vehicles along multiple path segments in near real-time even though this is not necessary for the proper functioning of NPSNET-MES. This is highly desirable when introducing active players into the path planning problem.

The path generation module produces paths for multiple vehicles, but only displays the lead vehicle's future path. Figures 8 shows an example of the SAF controller's input path and Figure 9 depicts the output from the NPSNET-MES path generation module. The path found by the path generation module is not necessarily the most optimal path in terms of distance. The path is direct and the path generation module does not need to

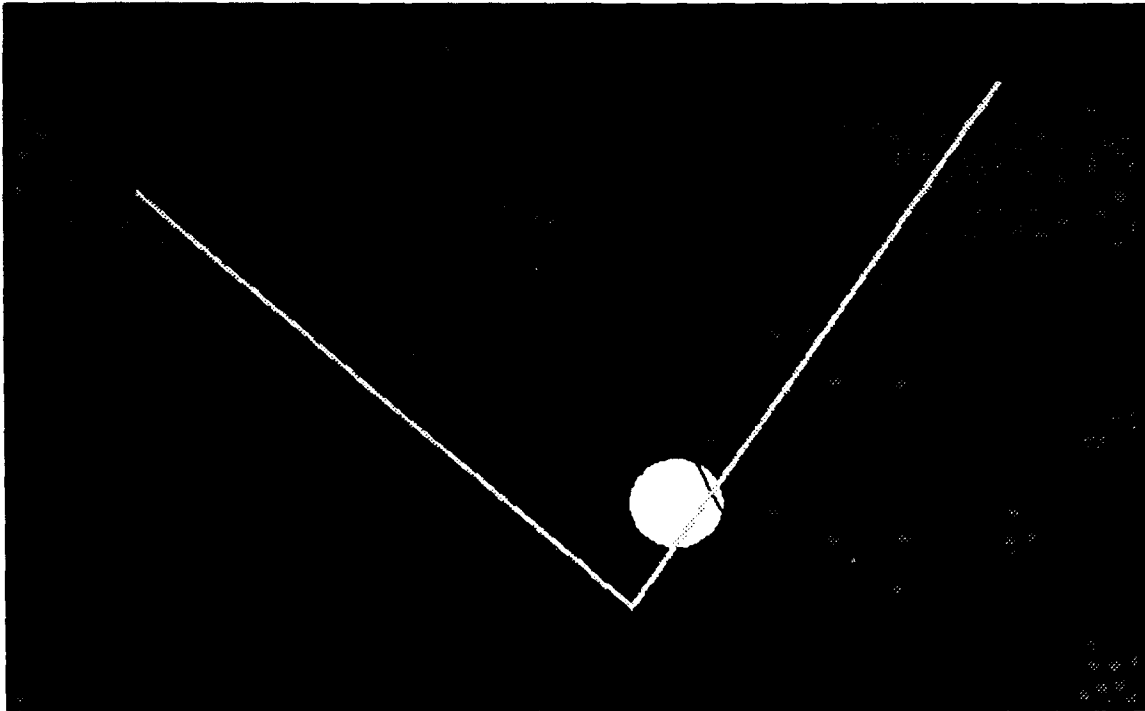


Figure 8. SAF Controller Input Path

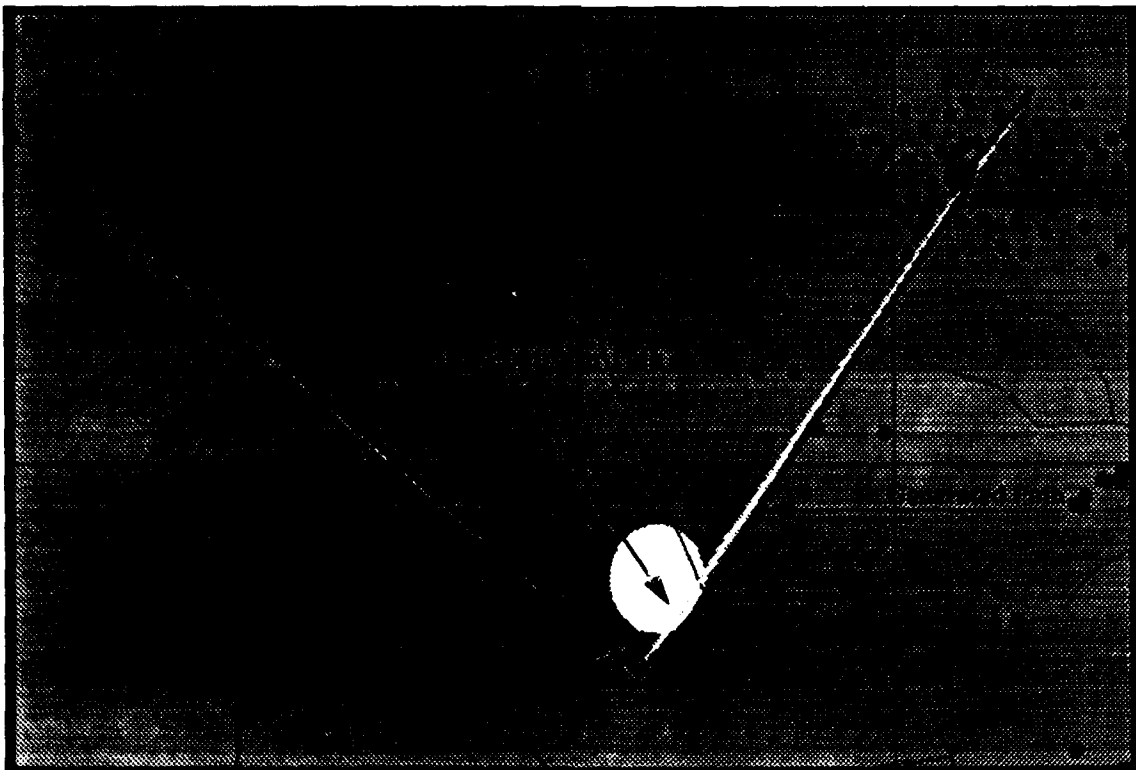


Figure 9. SAF Controller Path and Generated Path

search the entire database to find a solution. For the purposes of a simulation, the path generated is likely to be analogous to a path found by the humans that the path generation module seeks to emulate.

B. VEHICLE CONTROLLER MODULE

The vehicle controller module provides adequate control of the SAF vehicles employed in NPSNET. The controller sends real-time update messages to NPSNET through NPSNET-NET. The system takes the default file, **pathfile.dat**, or the user specified file from the command line, to use for path data. The system is operational and sending out the first packet in less than 5 seconds. The set-up time is an important consideration, since the SAF controller introduces the SAF into the simulation at a specific time.

The path points that the path generation module finds are exact locations. However, the system must communicate over a network and use a system clock that sends messages over the network at a rate of one per 10 milliseconds. These two factors combine to introduce an error margin of less than one percent in the SAF vehicle's projected and actual paths. Table 1 shows the snap distance caused by the vehicle being late to a turn. To correct this deficiency, the vehicle controller system sends the X-Z coordinate for the start of each path segment, so the vehicle has a fresh start and there is no accumulated error factor. This deficiency shows up as noticeable shift in the vehicle position at each path segment turn. Without sending the next path segment point, the vehicle's actual path will not mirror the projected path. Therefore in a high obstacle density environment,

there is a high potential for the SAF vehicle running into an obstacle and self-destructing. To correct this problem, NPSNET-NET must be able to operate the millisecond rate of transmission and reception of data packets. The system is currently not able to perform correctly at this speed due to network limitations. Figure 10 shows a comparison of the projected and actual paths.

TABLE 1. SAF VEHICLE SNAP DISTANCE

speed/ delay	10ms	50ms	100ms
10km/hr	.03m	.14m	.28m
30km/hr	.08m	.42m	.83m
100km/hr	.27m	.83m	1.67m

C. SUMMARY OF RESULTS

NPSNET-MES is a useful tool for generating multiple SAFs to make the NPSNET simulation a worthwhile training vehicle. The SAF controller can prepare the SAF missions in advance of the simulation in a relatively short period. Once the simulation is ongoing, the SAF controller introduces SAF formations at anytime and anyplace the SAF controller designates. NPSNET-MES provides many new opportunities to enhance combat simulations, but there is much left to do in this research that the next chapter addresses.

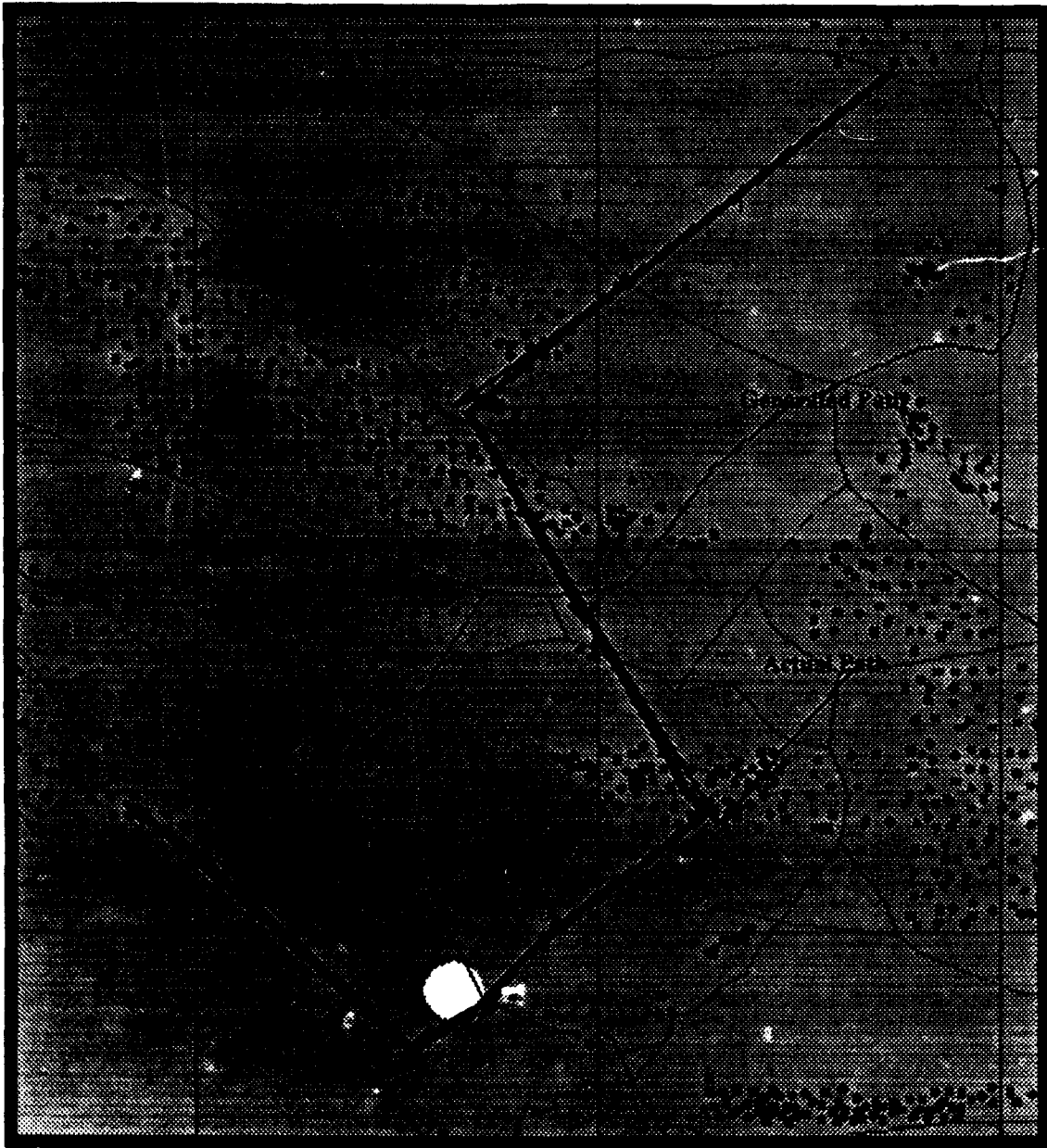


Figure 10. Actual SAF Vehicle Trace and Generated Path

VI. SUMMARY AND CONCLUSIONS

A. LIMITATIONS

1. Path Generation Module

NPSNET-MES path generation module is limited in the following ways:

- No dynamic path planning for the SAFs to react to other players during the simulation.
- Produces only one combat formation type for the entire mission.
- Terrain slope considerations are not incorporated in the path planning algorithm.

The most serious limitation with the system is the inability of the SAF to react to other players in the simulation. The SAF missions are pre-set before the simulation begins and cannot be altered once it commences. This was a design decision made at the outset of the project. The deficiency can be corrected by incorporating a local path generation capability within the vehicle controller module. When a SAF comes within range of an active player, the vehicle controller module path generation function would generate a local path around the moving obstacle and then the SAF reenters the previous path at the closest point.

The other limitations are important, but were not incorporated into NPSNET-MES by design. The path generation module places all follow-on vehicles in a column of wedges. This is a good movement formation, but there are many occasions where

other formations would be appropriate. This can be designed into the **convoy_disposition** function by giving the SAF controller some options during his path planning preparation.

Terrain slope considerations are not incorporated into the path generation module because the design called for a fast and efficient path planner. Terrain analysis requires more computation per path segment since the path generator evaluates each path segment terrain slope to determine go and no-go terrain. A quick solution to this problem is to make all terrain areas that have a no-go slope into obstacles and add them to obstacle database. For a more complete treatment of this problem see the Ross thesis [Ross, 1989, pp.22-79].

2. Vehicle Controller Module

NPSNET-MES vehicle controller module is limited in the following ways:

- Limited SAF vehicle reaction to active simulation players.
- Capable of networking only on the IRIS workstations subnetwork.
- Projected and actual path plots deviate due to clock speed and network transmission times.

The inability of the SAF vehicles to react to the active players is a limiting factor. This limitation is within the scope of the project, but a design decision was made early in the design phase to not implement multiple reaction capabilities because of the sheer size of the project. The SAF vehicles die when attacked because NPSNET-MES no longer sends update positions and reduces the speed to zero. By increasing the number of items that the vehicle controller module checks from the network, the reaction

capability can be upgraded. A simple fix is to have the SAF fire back at attacking vehicles.

The system is currently implemented for networking on the IRIS workstation subnet. Once NPSNET-NET is programmed to multicast outside of this subnet, the vehicle controller module can be recompiled for implementation on any system on the network. This is not a serious limitation and can easily be implemented.

The final limitation is not a serious one since the deviation is small and the shifting movement is not conspicuous. To fix the problem, the system must be able to operate at the millisecond rate or faster since the path points are in an ascending order queue. Some path points may have the same time stamp causing a delay for at least one of the SAF vehicles. NPSNET-NET is not able to effectively operate faster than its current rate due to hardware system limitations. The limitations create a bottleneck because there is only a single wire and single port on the Ethernet. A possible solution is to introduce multiple queues using IRIS's multiple CPU capability. There will always be some error due to transmission time delay, but this effect is negligible as long as the machines are in relative proximity.

B. AREAS FOR FURTHER RESEARCH

There are many potential applications for this research that can be explored. This field of research is new and the need for its products is increasing. The most important upgrade to this system is the enhanced capability of SAF reaction to other players in the simulation. The use of Intelligence Preparation of the Battlefield (IPB) is an all

encompassing analysis for mission planning. IPB takes a great deal more into consideration when planning a mission than just obstacle avoidance, which is the primary focus of this research. A more detailed terrain analysis determines route selection by including factors such as slope and terrain composition. The further incorporation of these elements makes the SAF mission planning more robust and thus more authentic.

C. SUMMARY

NPSNET-MES integrates semi-automated forces into a networking simulation and provides more realism into the battlefield scenario. The need to populate the battlefield with friend and foe forces makes NPSNET-MES an excellent tool. The SAF controller has great latitude in deciding the composition and mission of each SAF formation. The SAF controller also determines exactly when to introduce SAF into the NPSNET simulation.

The two components of NPSNET-MES provide a user friendly interface to allow the SAF controller to introduce the right mixture of semi-automated forces. The SAF controller uses the 2D map in the path generation module prior to the simulation to generate the SAF mission route. The vehicle controller module uses a workstation on the simulation network to send update packets containing current vehicle position, direction, and speed. The system is modular in design and can easily be the basic framework for more powerful SAF generators.

The best approach to incorporating SAF into NPSNET is to make the SAF work within the already existing framework of NPSNET. The use of NPSNET-NET greatly

facilitates the incorporation of the SAF movements into the NPSNET simulation. This work shows the use of a modified breadth-first search is a good approach to making a relatively fast and near optimal path planner. The good solution is more practical and more human-like than the best solution. Therefore, NPSNET-MES integrates SAF into NPSNET with good results.

D. CONCLUSIONS

NPSNET-MES fulfills a real need in networking simulations to populate the simulations with semi-automated forces. NPSNET-MES makes the NPSNET simulation a realistic training tool. NPSNET-MES replaces the use of randomly generated vehicles that have no real purpose other than populating the battlefield, with SAF that clearly have a mission and a realistic movement.

APPENDIX A. USER'S GUIDE

A. PATH GENERATION MODULE

This module must be run on an IRIS workstation. The executable command is 'demo'. The user can specify a path file name by typing the file name in after **demo** and beginning the file name with the letter 'p'. The default file that the system produces is **pathfile.dat**. The user sees the 2D data map after the start-up sequence concludes. During this portion the user can zoom in to a particular area by pressing the middle mouse button on the area desired. The NPSNET-MES displays the X and Z coordinates at the top of the screen in the information panel (Figure 11). The user then selects a function key, F1-F4, to zoom in on the desired area. The user zooms out by pressing the F1 key. Once the user decides where to plan the path and at what resolution, the user then presses the return key to enter the path planner.

NPSNET-MES will prompt the user for the number of combat formations. The user will use Dial 1 on the dial box to select the number of combat formations and then press return. The user then enters a loop to specify the variables for each combat formation. NPSNET-MES prompts the user for number of vehicles in the combat formation. At this point the user can option to specify SAF vehicle types and identification numbers. The Help Screen displays the user requirements as the user plans the mission.

The next task is to input the desired route for the SAF formation. The user starts this process once he clicks the right mouse button. The system will display a red rubber

band line from the origin of the first click to the current mouse location. The user can modify the start by repressing the right mouse button. The user then presses the left mouse button to set the path segment. The system will display that segment in green. The user can specify up to 500 path segments. The user can press the pause key to generate the proposed course. The user has three options at this time. First, he can select this path by pressing the return key. The user can modify the path by picking the path segment and moving a point. The final option is to press the escape key and start over again.

If the user opts to specify the vehicle types or identification numbers, the user inputs this information using Dial 2 for vehicle types and Dial 3 for identification numbers. The vehicle display screen will show the actual vehicles that the SAF controller selects. The user continues in this manner until all combat formations are finished.

B. VEHICLE CONTROLLER MODULE

This module requires very little user interface. This program must currently be run on an IRIS workstation, but soon will operate independent of the IRISs. The start-up command is **neto** and the user can specify a path file for execution or use the default file, **pathfile.dat**. The user must ensure that **netdemon** is running in the background for that workstation before executing **neto**.

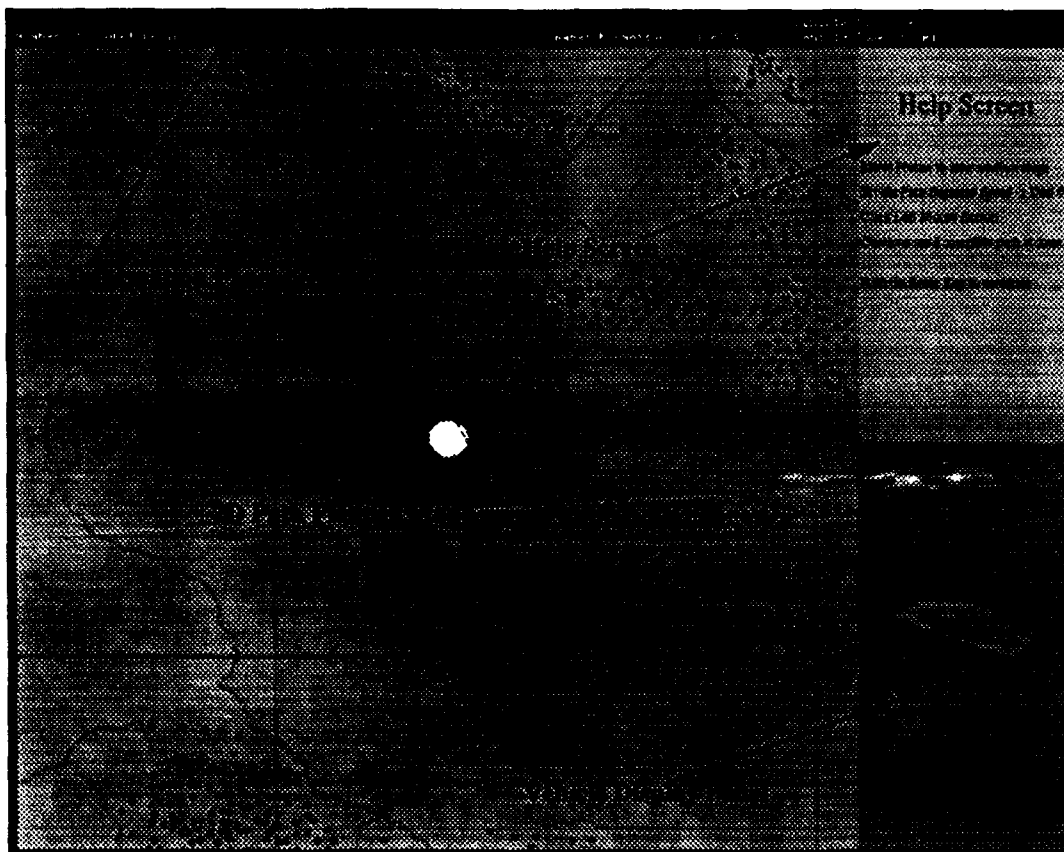


Figure 11. NPSNET-MES Display Panel

APPENDIX B. DATA STRUCTURES AND FUNCTION DESCRIPTIONS

A. DATA STRUCTURES

1. Path Generation Module

- **struct mapgridstruct mapgrid[NUMOFGRID][NUMOFGRID]** - Matrix containing elevation and obstacle list.
- **struct vehobjtype vehtypearray[MAXTYPEOFVEH]** - Array containing vehicle definitions by type and name.
- **struct pathtype full_path[MAXCONVOY][MAXPATH][MAXPTS]** - Matrix containing the all path points, direction, speed, distance, vehicle type, convoy, and vehno.
- **Point2 path_array[MAXPATH][MAXPTS]** - Matrix containing the path of points for the lead combat formation vehicle.
- **struct pathdef convoy_array[MAXCONVOY][MAXPATH][MAXPTS]** - Matrix containing the designated path Start Point, intermediate points, and Goal for each vehicle in the combat formation.
- **int vehicletype[MAXTYPEOFVEH]** - Array containing the indices of only valid vehicle types.
- **struct objdefsstruct objdefs[MAXTYPEOFVEH]** - Array containing name and radius of vehicle types.
- **struct pathtype obstacle_list[MAXPTS]** - Array containing list of obstacles already checked.
- **float total_distance[MAXCONVOY][MAXPATH]** - Array containing the total distance for each convoy by path.

2. Vehicle Controller Module

- **struct pathstruct current_ptr** - Pointer maintaining the current position of the singly linked list of vehicles sorted by time.
- **struct objdefsstruct objdefs[MAXTYPEOFVEH]** - Array containing name and radius of vehicle types.
- **struct commara *messbuff** - Message buffer data structure.

B. FUNCTION DESCRIPTIONS

The following are critical functions for path generation and vehicle controller modules:

1. Path Generation Module

- **draw_route(convoy)** - Draws a rubber band line between path points using the mouse and inputs those x, z coordinates into path_array[]. This function returns the number of points in the path.
- **convoy_disposition(convoy, numofpaths, numofvehs)** - Uses the path_array[] to assign vehicle formation disposition for the Start Point, designated intermediate points, and Goal using the lead vehicle as the point of alignment. Fills the convoy_array[] with x, z coordinates for all combat formation vehicles.
- **path_finder(numofpaths, convoy, vehno)** - Fills the full_path[] array with the x, z coordinates for points found during the path search using the algorithm described earlier. Determine the direction of travel to pass off to correct find_quad_N_path Returns the total number of points in the path.
- **find_quad_N_path(convoy, vehno, P1, P2)** - There are four (N=one, two, three, or four) quadrants that the path direction can follow. Indexes the mapgrid[][] array to reduce search space. Using the input points makes call to following functions to determine intersection of any obstacles and recursive call to its self until the entire path is found.
- **circle_intersection(P1, P2, Cc)** - Returns a Boolean of true or false if the obstacle is obstructing the proposed path.

- **find_closest_tangent(P1, P2, Cc)** - Returns the closest tangent point on the obstacle circle to the next designated point. This is an intermediate point if this obstacle is blocking the path.
- **add_path_segment(convoy, vehno, P2)** - Adds valid points to the full_path[] array and calculates the direction and places that direction into the array.
- **calculate_distance(convoy, vehno, path_seg_count)** - Using the points in the full_path[] calculates the distance between points and calculates total_distance[[]].
- **calculate_speed_and_time(convoy, vehno, path_seg_count, mes_speed)** - Using the distances calculated earlier fills the speed and time fields of the full_path[] array using the lead combat formation vehicle as a meter for the whole formation's relative velocity.
- **draw_full_path(convoy, vehno, pathno, path_seg_count)** - Draws the lead vehicles projected path using the array of points in full_path[].

2. Vehicle Controller Module

- **add_to_sorted_list(pathdata, convoy, vehno)** - Adds records to a singly linked list that is sorted by the time field that are read from the path file.
- **addproctonet(&mypid, &commnumber)** - Adds MES process to the network so that MES can communicate.
- **getpackets(commnumber, curtime, veharray, vehpos, drivenveh, masterflag)** - Gets packets from the Ethernet regarding a vehicle's status, such as alive or dead.
- **sendupdatemess(commnumber, curtime, veharray[convoy][vehno], vehno)** - Multicasts a message over the Ethernet with the changing vehicle statuses.
- **detachfromnet(mypid, commnumber)** - Detaches the MES process from the network.

LIST OF REFERENCES

Beaton, R.M., Adams, M.B., and Harrison, "Real-Time Mission and Trajectory Planning," *Proceedings of the 26th IEEE Conference on Decision and Control*, Volume 3 of 3, pp.1954-1959, 9-11 December 1987.

MacPherson, D.L., *A Computer Simulation Study of Rule-Based Control for an Autonomous Underwater Vehicle*, Master's Thesis, Naval Postgraduate School, Monterey, California, June 1988.

McConkle, C.M, and Nelson, A.H., *A Prototype Simulation System for Combat Vehicle Coordination and Visualization*, Master's Thesis, Naval Postgraduate School, Monterey, California, June 1988.

Ross, R.S., *Planning Minimum-Energy Paths in an Off-Road Environment with Anisotropic Traversal Costs and Motion Constraints*, PhD Dissertation, Naval Postgraduate School, Monterey, California, June 1989.

Shannon, L.R., and Teter, W.A., *An Autonomous Platform Simulator (APS)*, Master's Thesis, Naval Postgraduate School, Monterey, California, June 1989.

Zyda, M.J., McGhee, R.B., McConkle, C.M., Nelson, A.H., and Ross, R.S., "A Real-Time, Three-Dimensional Moving Platform Visualization Tool," *Computer & Graphics*, Vol. 14, No. 2, pp.321-333, 1990.

Zyda, M.J., and Pratt, D.R., "NPSNET, A 3D Visual Simulator for Virtual World Exploration and Experimentation," *SID 91 Digest*, pp.361-363, May 1991.